ELSEVIER

Theoretical Computer Science 179 (1997) 301–317

Theoretical Computer Science

On reachability equivalence for BPP-nets

Hsu-Chun Yen

Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan 106, Republic of China

Received April 1995; revised February 1996
Communicated by O.H. Ibarra

Abstract

In this paper, we study the complexity of the reachability equivalence problem for BPP-nets. BPP-nets are closely related to *Basic Parallel Processes*, which form a subclass of Milner's CCS. We show the reachability equivalence problem for BPP-nets to be solvable in $DTIME(2^{2^{ds^3}})$, where d is a constant and s is the size of the problem instance, when a standard binary encoding scheme is used. To that end, we provide a new characterization for computations in BPP-nets, which, in turn, facilitates the derivation of small semilinear set representations for the reachability sets of BPP-nets. As for the lower bound, the problem is shown to be Π_2^P*-hard*. Our results improve upon the previous decidability result of the reachability equivalence problem for BPP-nets.

1. Introduction

BPP-nets provide a net semantics for *Basic Parallel Processes* (*BPP*), which form a subclass of Milner's CCS (see, e.g., [1, 3, 7]). Simply speaking, a BPP-net is a Petri net in which each transition has exactly one input place, and the firing of a transition removes exactly one token from its input place [4, 5]. It seems, on the surface, that the computational power of BPP-nets is rather limited. The limitation is a direct consequence of the inability for BPP-nets to model 'synchronization' actions, which require places to synchronize through transition firings. (This is why such Petri nets are the so-called *communication-free nets* [6].)

What makes BPP-nets theoretically interesting, aside from their close connection to BPPs, includes the following. First, BPP-nets are also computationally equivalent to the so-called *commutative context-free grammars* defined and investigated in [10, 11]. Of many problems considered in [10], the *uniform word problem* was shown to be solvable in NP through a somewhat complicated proof. In a recent article [4], an alternative and simpler proof has been given for the NP upper bound of the uniform word problem, taking advantage of the connection between BPP-nets and commutative context-free grammars as well as the fact that the reachability problem for BPP-nets

0304-3975/97/$17.00
PII S0304-3975(96)00147-8

is solvable in NP. Second, surprising results have been shown regarding the issue of deciding equivalence for labeled BPP-nets with respect to various equivalence notions defined in the *linear time/branching time hierarchy* of [17]. Deciding *bisimulation equivalence* has been shown to be decidable [2]. However, for all the equivalences of the *linear time/branching time hierarchy* below bisimulation equivalence, deciding equivalence turns out to be undecidable [8]. The undecidability result is somewhat surprising, taking into consideration the rather limited computational power of BPP-nets. As for reachability equivalence (which coincides with the conventional equivalence of Petri net reachability sets), it has recently been shown in [4] that BPP-nets always exhibit effective semilinear reachability sets, thus yielding a decidability result.

Motivated by the work (in particular, the technique) of [4], in this paper we develop a new characterization for paths in BPP-nets. As we will see later, the simple structure of circuits in BPP-nets plays a crucial role in our analysis. (A *circuit* of a Petri net is simply a closed path (i.e., a cycle) in the Petri net graph.) By and large, the presence of complex circuits, in general, is troublesome in Petri net analysis. In fact, strong evidence has suggested that circuits constitute the major stumbling block in the analysis of Petri nets. To get a feel for why this is the case, it is well known that in a Petri net $\mathscr{P}$ with initial marking μ_0, a marking μ is reachable (from μ_0) in $\mathscr{P}$ *only if* there exists a column vector $x \in \mathrm{N}^k$ such that $\mu_0 + A \cdot x = \mu$, where k is the number of transitions in $\mathscr{P}$ and A is the *addition matrix* of $\mathscr{P}$. The converse, however, does not necessarily hold. In fact, lacking a simple necessary and sufficient condition for reachability in general has been blamed for the high degree of complexity in the analysis of Petri nets. (Otherwise, one could tie the reachability analysis of Petri nets to the *integer linear programming* problem, which is relatively well understood.) There are restricted classes of Petri nets for which necessary and sufficient conditions for reachability are available. Most notable, of course, is the class of *circuit-free* Petri nets (i.e., Petri nets without circuits) for which the equation $\mu_0 + A \cdot x = \mu$ is sufficient and necessary to capture reachability. A slight relaxation of the circuit-freedom constraint yields the same necessary and sufficient condition for the class of Petri nets without *token-free* circuits in every reachable marking [16]. By taking advantage of simple circuits offered by BPP-nets, in this paper we show that any path in a BPP-net can be rearranged into some canonical form, which, in turn, facilitates the derivation of 'small' semilinear set representations for BPP-nets. This result, in conjunction with a known result concerning the complexity of deciding equivalence for semilinear sets presented in [9, 12], yields a $DTIME(2^{2^{d*s^3}})$ [1] upper bound of the reachability equivalence problem for BPP-nets, where s is the size of the problem instance (when a standard binary encoding scheme is used) and d is a fixed constant.

The contributions of this paper include the following. Our $DTIME(2^{2^{d*s^3}})$ result improves upon the previous decidability result presented in [4]. (In [4], the decidability

[1] $DTIME(f(n))$ represents the class of languages accepted by deterministic Turing machines using at most $f(n)$ time.

result was obtained by showing the reachability sets of BPP-nets to be effectively semilinear. The work, however, did not reveal any complexity bounds for the reachability equivalence problem.) As for the lower bound, at this moment we are able to show the problem to be Π_2^P-hard [2], directly following a result presented in [11] concerning the complexity of the equivalence problem for commutative context-free grammars. As a by-product, our analysis yields yet another proof for the NP upper bound of the reachability problem for BPP-nets. (We show that checking the reachability property for BPP-nets is tantamount to solving an integer linear programming problem. The approach used in [4], on the other hand, requires that certain structure (called *siphon*) of Petri nets be examined.) Finally, we feel that the new characterization for paths in BPP-nets is interesting in its own right, and may have other applications to the analysis of Petri nets.

The remainder of this paper is structured as follows. In Section 2, we formally define the model of Petri nets, the reachability equivalence problem, and the notations used throughout this paper. In Section 3, we show that BPP-net computations can always be rearranged into some canonical form, facilitating the use of integer linear programming to solve the reachability problem. Finally, in Section 4, we derive small semilinear set representations for BPP-nets, which, in turn, give rise to an upper bound for the reachability equivalence problem.

2. Preliminaries

Let Z (N) denote the set of (nonnegative) integers, and Z^k (N^k) the set of vectors of k (nonnegative) integers. For a k-dimensional vector v, let $v(i), 1 \leqslant i \leqslant k$, denote the ith component of v. For a $k \times m$ matrix A, let $a_{i,j}$, $1 \leqslant i \leqslant k, 1 \leqslant j \leqslant m$, denote the element in the ith row and the jth column of A, and let a_j denote the jth column of A. For a given value of k, let $\mathbf{0}$ denote the vector of k zeros (i.e., $\mathbf{0}(i) = 0$ for $i = 1, \ldots, k$). We let $|S|$ be the number of elements in set S. Given a column vector x, we let x^{T} (which is a row vector) denote the *transpose* of x. Given an alphabet (i.e., a finite set of symbols) Σ, we write Σ^* to denote the set of all finite-length strings (including the empty string λ) using symbols from Σ. We write Σ^+ to denote $\Sigma^* - \{\lambda\}$.

A *Petri net* (*PN*) is a triple (P, T, φ), where P is a finite set of *places*, T is a finite set of *transitions*, and φ is a *flow function* $\varphi : (P \times T) \cup (T \times P) \rightarrow N$. In this paper, k and m will be reserved for $|P|$ (the number of places in P) and $|T|$ (the number of transitions in T), respectively. A *marking* is a mapping $\mu : P \rightarrow N$. A transition $t \in T$ is *enabled* at a marking μ iff for every $p \in P$, $\varphi(p,t) \leqslant \mu(p)$. A transition t may *fire* at a marking μ if t is enabled at μ. We then write $\mu \xmapsto{t} \mu'$, where $\mu'(p) = \mu(p) - \varphi(p,t) + \varphi(t,p)$ for all $p \in P$. A sequence of transitions $\sigma = t_1 \ldots t_n$ is a *firing sequence* from μ_0 iff $\mu_0 \xmapsto{t_1} \mu_1 \xmapsto{t_2} \cdots \xmapsto{t_n} \mu_n$ for some sequence of markings $\mu_1, \ldots, \mu_n$. (We also write '$\mu_0 \xmapsto{\sigma} \mu_n$'.) We write '$\mu_0 \xmapsto{\sigma}$' to denote that σ is enabled

[2] Π_2^P denotes the set of all languages whose complements are in the second level of the polynomial-time hierarchy [15].

and can be fired from μ_0, i.e., $\mu_0 \xmapsto{\sigma}$ iff there exists a marking μ such that $\mu_0 \xmapsto{\sigma} \mu$. The notation $\mu_0 \xmapsto{*} \mu$ is used to denote the existence of a σ such that $\mu_0 \xmapsto{\sigma} \mu$. A *marked* PN is a pair $((P,T,\varphi),\mu_0)$, where (P,T,φ) is a PN, and μ_0 is a marking called the *initial marking*. Throughout the rest of this paper, the word 'marked' will be omitted if it is clear from the context. By establishing an ordering on the elements of P and T (i.e., $P=\{p_1,\ldots,p_k\}$ and $T=\{t_1,\ldots,t_m\}$), we define the $k\times m$ *addition matrix* A of (P,T,φ) so that $a_{i,j}=\varphi(t_j,p_i)-\varphi(p_i,t_j)$. Thus, if we view a marking μ as a k-dimensional column vector in which the ith component is $\mu(p_i)$, each column a_j of A is then a k-dimensional vector such that if $\mu \xmapsto{t_j} \mu'$, then $\mu'=\mu+a_j$. Let $\mathscr{P}=((P,T,\varphi),\mu_0)$ be a PN. The *reachability set* of $\mathscr{P}$ is the set $\mathrm{R}(\mathscr{P})=\{\mu \mid \mu_0 \xmapsto{\sigma} \mu$ for some $\sigma\in T^*\}$. The *reachability equivalence problem* (or simply *equivalence problem*) is that of determining, given two PNs $\mathscr{P}_1$ and $\mathscr{P}_2$ with the same set of places, whether $\mathrm{R}(\mathscr{P}_1)=\mathrm{R}(\mathscr{P}_2)$.

For ease of expression, the following notations will be used extensively throughout the rest of this paper. (Let σ,σ' be transition sequences, p be a place, and t be a transition.)

- $\#_\sigma(t)$ represents the number of occurrences of t in σ. (For convenience, we sometimes treat $\#_\sigma$ as an m-dimensional vector assuming that an ordering on T is established ($|T|=m$).)
- $\Delta(\sigma)=A\cdot\#_\sigma$ defines the *displacement* of σ. (Notice that if $\mu \xmapsto{\sigma} \mu'$, then $\Delta(\sigma)=\mu'-\mu$.) For a place $p\in P$, we write $\Delta(\sigma)(p)$ to denote the component of $\Delta(\sigma)$ corresponding to place p.
- $Tr(\sigma)=\{t \mid t\in T, \#_\sigma(t)>0\}$, denoting the set of transitions used in σ.
- $|\sigma|$ is the number of transitions in σ, i.e., $|\sigma|=n$ if $\sigma=t_1\ldots t_n$.
- $\sigma \dot{-} \sigma'$ is defined inductively as follows. Suppose $\sigma'=t_1\ldots t_n$. Let σ_0 be σ. If t_i is in σ_{i-1}, let σ_i be σ_{i-1} with the leftmost occurrence of t_i deleted; otherwise, let $\sigma_i=\sigma_{i-1}$. Finally, let $\sigma \dot{-} \sigma'=\sigma_n$. For example, if $\sigma=t_1t_2t_3t_4t_5$ and $\sigma'=t_4t_3t_1$, then $\sigma \dot{-} \sigma'=t_2t_5$. Intuitively, $\sigma \dot{-} \sigma'$ represents the transition sequence resulting from removing each transition of σ' from the leftmost occurrence of such a transition in σ (if the transition exists).
- $p^\bullet=\{t \mid \varphi(p,t)\geqslant 1, t\in T\}$ is the set of output transitions of p;
  $t^\bullet=\{p \mid \varphi(t,p)\geqslant 1, p\in P\}$ is the set of output places of t.
- ${}^\bullet p=\{t \mid \varphi(t,p)\geqslant 1, t\in T\}$ is the set of input transitions of p;
  ${}^\bullet t=\{p \mid \varphi(p,t)\geqslant 1, p\in P\}$ is the set of input places of t.

Notice that if $\mu_0 \xmapsto{\sigma} \mu$, then $\mu_0+A\cdot\#_\sigma=\mu$. (The converse, however, does not necessarily hold.) Given a path $\mu \xmapsto{\sigma} \mu'$, a sequence σ' is said to be a *rearrangement* of σ if $\#_\sigma=\#_{\sigma'}$ and $\mu \xmapsto{\sigma'} \mu'$.

A PN $((P,T,\varphi),\mu_0)$ is said to be a *BPP*-net [4] if

(1) $\forall t\in T, |{}^\bullet t|=1$, (i.e., every transition has exactly one input place), and

(2) $\forall p\in P, t\in T, \varphi(p,t)\leqslant 1$ (i.e., every arc going from a place to a transition has weight 1).

A *circuit* of a PN is a 'simple' closed path in the PN graph. (By 'simple' we mean all nodes are distinct along the closed path.) It is important to note that every circuit

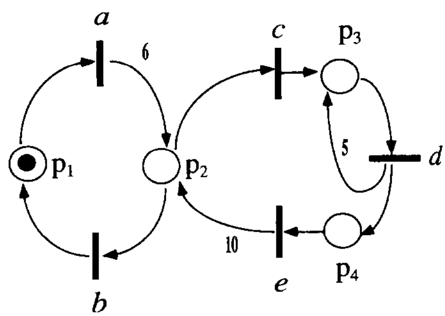

Fig. 1. A BPP-net.

$c = p_1t_1p_2t_2\cdots p_nt_np_1$ in a BPP-net must have ${}^\bullet t_i = \{p_i\}$, for every $i, 1 \leqslant i \leqslant n$. See Fig. 1 for an example of a BPP-net. (Notice that the firing of a transition may deposit more than one token into a place. In Fig. 1, for example, the firing of transition e adds 10 tokens to place p_2.) Given a circuit $c = p_1t_1p_2t_2\cdots p_nt_np_1$, let $P_c = \{p_1, p_2, \cdots, p_n\}$ denote the set of places in c. (With a slight abuse of notation, we sometimes use c to denote transition sequence $t_1t_2\cdots t_n$ of circuit $p_1t_1p_2t_2\ldots p_nt_np_1$ when places are not important.) We define the token count of circuit c in marking μ to be $\mu(c) = \sum_{p\in P_c}\mu(p)$. A circuit c is said to be *token-free* in μ iff $\mu(c) = 0$. A set of circuits $\mathscr{C}=\{c_1,c_2,\ldots,c_n\}$ is said to be *connected* iff for every i,j, $1\leqslant i,j\leqslant n$, there exist $1\leqslant h_1,h_2,\ldots,h_r\leqslant n$, for some r, such that $h_1 = i$, $h_r = j$, and for every $1\leqslant l < r$, $P_{c_{h_l}}\cap P_{c_{h_{l+1}}} \neq \emptyset$. In words, every pair of neighboring circuits in the sequence $c_{h_1},c_{h_2},\ldots,c_{h_r}$ share at least one place. For a simple circuit c, we also use $\#_c$ to denote the vector count of transitions used in c, i.e., $\#_c(i) = 1$ if t_i is in c; $\#_c(i) = 0$, otherwise. A sequence σ is said to *cover* circuit c if $\#_c\leqslant\#_\sigma$, i.e., every transition of c appears in σ.

For a vector $v \in N^k$ and a finite set ρ $(= \{v_1,\ldots,v_n\}$, for some $n) \subseteq N^k$, the set $\mathscr{L}(v,\rho) = \{v|\exists a_1\ldots,a_n \in N, v = v + \sum_{i=1}^{n} a_i * v_i\}$ is called the *linear set* with *base* v over the set of *periods* ρ. A *semilinear set* is a finite union of linear sets.

To deal with the complexity issue, it is essential to define the *sizes* of Petri nets and semilinear sets in a precise manner. Throughout this paper, each integer will be represented by its binary representation. The *length* of an integer is the number of bits of its binary representation. The size of a set (or vector) of integers is defined to be the sum of the lengths of the components. The size of a linear set $\mathscr{L}(v,\rho)$ is the size of vector v plus the sum of the sizes of the vectors in ρ. The size of a semilinear set is the sum of the sizes of its constituent linear sets. Consider a Petri net $\mathscr{P}=((P,T,\varphi),\mu_0)$, where $P = \{p_1,\ldots,p_k\}$ and $T = \{t_1,\ldots,t_m\}$. Each transition $\varphi(p_i,t_j) = d$ $(\varphi(t_j,p_i) = d)$ can be thought of as a four tuple $(0,i,j,d)$ $((1,j,i,d))$. (The first component (0 or 1) is to indicate the flow direction (0: from a place to a transition; 1: from a transition to a place). In this way, φ can be treated as a set of four tuples. Now the size of Petri net $\mathscr{P}$ can be defined as $\lceil\log_2 k\rceil + \lceil\log_2 m\rceil +$ the sum of the sizes of elements in φ + the size of μ_0. Since the binary representation is used, the firing of a transition may result in removing (or adding) 2^s tokens from (to) a place, where s is the size of the Petri net.

For more about Petri nets and their related problems, see [13, 14].

3. Canonical paths in BPP-nets

To derive the complexity of the equivalence problem, we begin with a few lemmas which are important in characterizing computations in BPP-nets. In the literature, one of the few techniques proven to be useful for analyzing PNs relies on the ability to rearrange PN paths into some 'canonical' form. As one might expect, the simple structure of circuits in BPP-nets (in particular, the ability to repeat a circuit for an arbitrary number of times at any marking at which the circuit is marked) suggests a good starting point for devising a rearrangement technique. The first attempt, perhaps, is to fire a circuit immediately when one of its transitions becomes enabled, even though the transitions of the circuit are interleaved with others in the original path. Unfortunately, such an attempt does not work as Fig. 1 indicates. (In Fig. 1, '$acdeb$' is a legal firing sequence, whereas 'ab(any permutation of cde)' is not.) To circumvent such a difficulty, we first present a nice property concerning any set of connected circuits in BPP-nets.

Lemma 1. *Let $\mathscr{C}=\{c_1,c_2,\ldots,c_n\}$ be a set of connected circuits in a BPP-net $\mathscr{P}$ and μ be a marking with $\mu(c_i) > 0$, for some i. For arbitrary integers $a_1,a_2,\ldots,a_n > 0$, there exists a sequence σ such that $\mu \overset{\sigma}{\longmapsto}$ and $\#_\sigma = \sum_{j=1}^{n} a_j(\#_{c_j})$. (In words, from μ there exists a firable sequence σ utilizing circuit c_j exactly a_j times, for every j.)*

Proof. Without loss of generality, we assume $i = 1$, and let p_1 be a place in c_1 such that $\mu(p_1) > 0$. The proof is done by induction on the number of circuits in $\mathscr{C}$. To help explain the proof, see Fig. 2.

Induction Basis: For $n = 1$, the result is quite obvious. (The sequence $t_1 \cdots t_r$ in Fig. 2 can be fired an arbitrary number of times.)

Induction Hypothesis: Assume that the assertion is true for $n \leqslant h$.

Induction Step: Consider $n = h + 1$. Starting from place p_1, let $p_2, \ldots, p_r$, for some r, be places along c_1. Let $\mathscr{C}_j$ $(1 \leqslant j \leqslant r)$ be the *largest* connected subset of $\mathscr{C} - \{c_1\} - (\bigcup_{0 \leqslant l \leqslant j-1} \mathscr{C}_l)$ in which one of its circuits contains place p_j. (Notice that $\mathscr{C}_j$ might be empty.) See Fig. 2. By induction hypothesis, all circuits in $\mathscr{C}_j$ can be fired arbitrarily, provided that p_j is marked. Let t_i $(1 \leqslant i \leqslant r)$ be the transition from places p_i to p_{i+1} along circuit c_1 (assuming that $p_{r+1} = p_1$). Then the desired sequence σ is the following: (sequence guaranteed by induction hypothesis for $\mathscr{C}_1$) t_1 (sequence guaranteed by induction hypothesis for $\mathscr{C}_2$) $\cdots t_{r-1}$ (sequence guaranteed by induction hypothesis for $\mathscr{C}_r$) t_r $(t_1 \cdots t_r)^{a_1-1}$. □

The idea of rearranging an arbitrary path in a BPP-net into a 'canonical' one is as follows. To give the reader a better feel for such a rearrangement, we accompany our

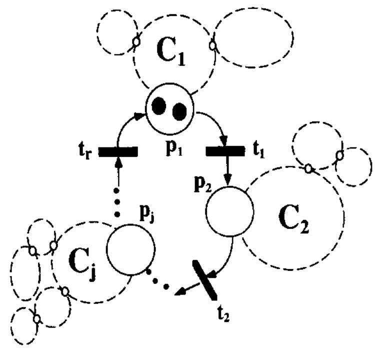

Fig. 2. A set of connected circuits.

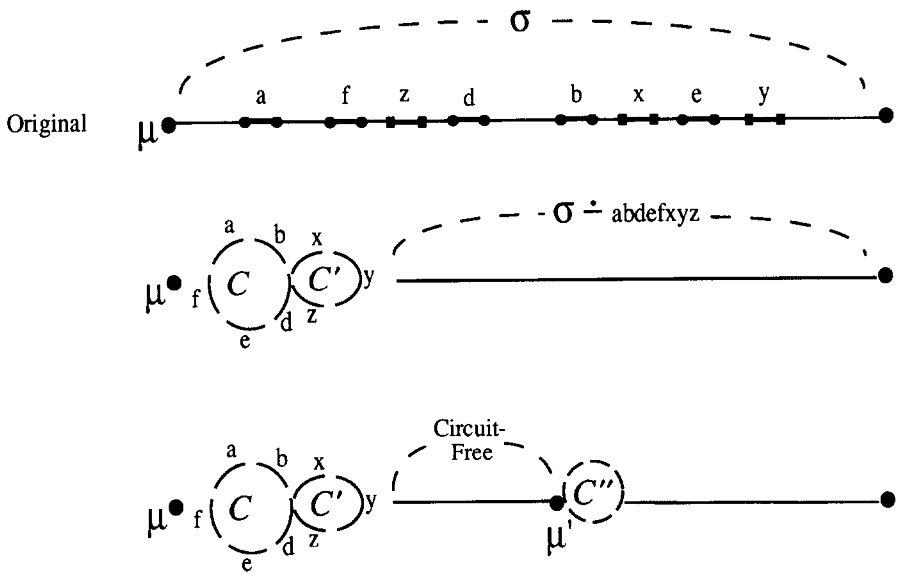

Fig. 3. Rearranging a path in a BPP-net into a canonical one.

subsequent discussion with Fig. 3. Suppose $\mu \overset{\sigma}{\longmapsto}$ is a path, and c is a circuit covered by σ such that $\mu(c) > 0$. (In Fig. 3, circuit c consists of transitions a, b, d, e, and f.) Then we use c as a 'seed' to grow the *largest* collection of connected circuits that are covered by σ (for example, circuits c and c' in Fig. 3). We then follow a 'short' circuit-free transition sequence of the remaining path until reaching a marking in which a non-token-free circuit (with respect to the current marking) which is covered by the subsequent path exists. (See marking μ' and circuit c'' in Fig. 3.) Using such a newly found circuit as a new seed and repeating the above procedure, we are able rearrange an arbitrary path of a BPP-net into a 'canonical' one as the following lemma indicates. Notice that the above procedure need not be repeated for more than m times, because for each of the circuits collected in a marking, at least one of its transitions must be absent from the remaining path.

It is important to point out that the above rearrangement procedure is merely 'conceptual'. That is, we do not actually carry out the above procedure in the derivation of our complexity result. What the rearrangement concept does is that it suggests the existence of a canonical computation, upon which our derivation of semilinear set representations for BPP-nets relies.

Lemma 2. *Let μ be a reachable marking in a BPP-net $\mathscr{P} = ((P, T, \varphi), \mu_0)$. Then there exists a sequence $\sigma = \pi_1\alpha_1\pi_2\alpha_2 \cdots \pi_h\alpha_h$ $(1 \leqslant h \leqslant m,\ \alpha_i, \pi_i \in T^*)$ which witnesses $\mu_0 \xmapsto{\sigma} \mu$ and satisfies the following conditions:*

1. $\forall i, 1 \leqslant i \leqslant h$,

(a) *there exists a set $\mathscr{C}_i = \{c_1^i, \ldots, c_{r_i}^i\}$ $(r_i \leqslant m)$ of connected circuits such that*

$$\Delta(\pi_i) = \sum_{j=1}^{r_i} a_j^i \Delta(c_j^i) \quad \text{for some integers } a_1^i, a_2^i, \ldots, a_{r_i}^i > 0,$$

(b) *the remaining sequence $\alpha_i \cdots \pi_h\alpha_h$ does not cover any circuit which shares some place with circuits in $\mathscr{C}_i$, and*

(c) $\sum_{i=1}^{h} |\mathscr{C}_i| \leqslant m$, *i.e., the total number of distinct circuits considered above is bounded by the number of transitions of the PN.*

2. $\forall i, 1 \leqslant i \leqslant h-1$,

(a) $\#_{\alpha_i}(t) \leqslant 1, \forall t \in T$ *(in words, all transitions in α_i are distinct),*

(b) $\Delta(\alpha_i)(p) \leqslant 1, \forall p \in P$ *(in words, α_i removes at most one token from any place), and*

(c) *α_i is circuit-free (i.e., it does not cover any circuit).*

3. *α_h is circuit-free. Notice that α_h may contain multiple copies of a transition.*

Proof. We begin by proving the following claim which tells how 'cut-and-paste' technique can be applied to BPP-nets.

Claim. *Consider a path $\mu_1 \xmapsto{\sigma} \mu_2$. Let $\mathscr{C} = \{c_1, c_2, \ldots, c_z\}$ be a set of connected circuits and $a_1, a_2, \ldots, a_z$ be positive integers such that*

(a) $(\exists i, 1 \leqslant i \leqslant z)$ $(\mu_1(c_i) > 0)$ *(i.e., c_i is not token-free in marking μ_1)*

(b) $\sigma \dot{-} (c_1^{a_1} \cdots c_z^{a_z})$ *does not cover any circuit that shares some place with circuits in $\mathscr{C}$, and*

(c) $\sum_{j=1}^{z} a_j(\#_{c_j}) \leqslant \#_\sigma$.

Then there exist δ_1 and δ_2 such that

(1) $\#_{\delta_1} = \sum_{j=1}^{z} a_j(\#_{c_j})$,

(2) $\#_{\delta_2} = \#_{\sigma \dot{-} \delta_1}$, *and*

(3) $\mu_1 \xmapsto{\delta_1} \mu_3 \xmapsto{\delta_2} \mu_2$, *for some μ_3.*

(In words, σ can be rearranged into $\delta_1\delta_2$ such that δ_1 consists of the largest collection of connected circuits with at least one of them marked in μ_1.)

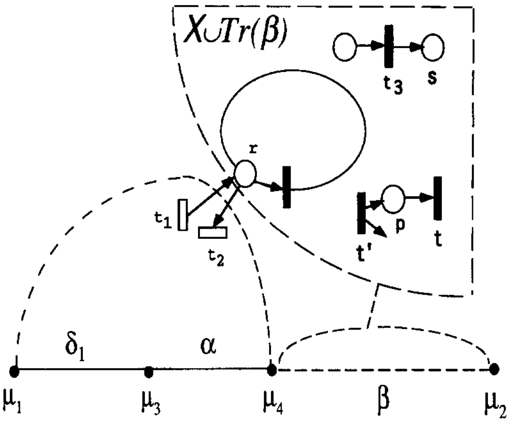

Fig. 4. A picture describing the concept used in the proof of Lemma 2.

To prove our claim, first notice that $\mu_1 \xrightarrow{\delta_1} \mu_3$ is guaranteed by Lemma 1; it suffices to prove that $\mu_3 \xrightarrow{\delta_2} \mu_2$, for some δ_2 which is a rearrangement of $\sigma \dot{-} \delta_1$. Suppose, to the contrary, that none of the permutations of $\sigma \dot{-} \delta_1$ is firable in μ_3. We let α be a *longest* sequence such that $\#_\alpha < \#_{\sigma \dot{-} \delta_1}$ and $\mu_3 \xrightarrow{\alpha} \mu_4$, for some μ_4. (By 'longest' we mean that for all α' with $\#_{\alpha'} < \#_{\sigma \dot{-} \delta_1}$ and $\mu_3 \xrightarrow{\alpha'}$, it must be the case that $|\alpha'| \leqslant |\alpha|$.) Let $\beta = (\sigma \dot{-} \delta_1) \dot{-} \alpha$. We let X be $\{p \mid {}^\bullet t = \{p\}, t \in Tr(\beta)\}$, i.e., X consists of all the input places of transitions in $Tr(\beta)$. Clearly, $\mu_4(p) = 0, \forall p \in X$. See Fig. 4. We now make the following observations:

1. $\forall p \in X, \exists t' \in Tr(\beta)$, such that $p \in t'^\bullet$. (This is because $\mu_4(p) + \Delta(\beta)(p) = \mu_2(p) \geqslant 0$ and $\mu_4(p) = 0$.)

2. there must be some place r in X such that either (i) $\mu_1(r) > 0$, *or* (ii) $(\exists t_1 \in Tr(\delta_1\alpha))$ $(\exists t_2 \in Tr(\delta_1\alpha))$ such that $(r \in t_1^\bullet)$ and ${}^\bullet t_2 = \{r\}$. (If neither (i) nor (ii), then none of the transitions in $Tr(\beta)$ could be fired in the original sequence σ. The existence of a t_2 results from $\mu_4(r) = 0$.)

Let R be the set of all places r satisfying Observation 2(i) or (ii) above. What we need next is to show that at least one place in R must be along a circuit consisting of some places in X and some transitions in $Tr(\beta)$. Suppose, to the contrary, that none of R is on a circuit; then there must be an $s \in R$ such that s cannot be reached from the remaining places in R through places in X and transitions in $Tr(\beta)$. For s, let t_3 be a transition guaranteed by Observation 1 above. Due to the selection of s, t_3 could never have been fired in σ since its input place would never possess a token (because the input place of t_3 (i.e., ${}^\bullet t_3$) is not in R, and none of R is capable of supplying a token to ${}^\bullet t_3$ directly or indirectly) – a contradiction. Intuitively, one can think of R as places through which tokens are 'pumped' into the sub-PN consisting of places in X and transitions in $Tr(\beta)$.

Let $r \in R$ be a place on a circuit, and t_2 (whose existence is guaranteed by Observation 2) be a transition in $\delta_1\alpha$ removing a token from r. If t_2 is in δ_1 (which comprises only circuits from $\mathscr{C}$), then c must have shared some place with one of the circuits in $\mathscr{C}$ – violating Assumption (b) of the claim. If t_2 is in α, then r is marked during the course of the path α, which implies that c should have been added to α – violating the assumption about α being the longest. This completes the proof of the claim.

In what follows, we only show how π_1 and α_1 are constructed; the remaining sequences can be obtained similarly. Suppose σ covers a circuit c which is not token-free in μ_0 (i.e., $\mu_0(c) > 0$). Initially, let $\mathscr{C}_1 = \{c_1^1\} = \{c\}$. (If σ does not cover any circuit marked in μ_0, then π_1 is empty.) The associated integer a_1^1 is the maximum number of occurrences of c in σ, i.e., $a_1^1(\#_c) \leqslant \#_\sigma$ but $a(\#_c) \nleqslant \#_\sigma, \forall a > a_1^1$. Let σ' be the resulting sequence of removing a_1^1 copies of c from σ. That is, $\sigma' = \sigma \dot{-} \overbrace{c \cdots c}^{a_1^1}$. It is important to notice that at least one of c's transitions is no longer in existence in σ'. The next step is to find, if one exists, a circuit c' which shares some place with at least one of $\mathscr{C}_1$; then add c' to $\mathscr{C}_1$ (i.e., $c_2^1 = c'$) and remove a_2^1 copies of c' from σ', where a_2^1 is the maximum number of occurrences of c' in σ'. Upon the completion of the above, at least one more transition becomes absent in the remaining sequence. Repeat the above procedure (at most m times) until no more circuit can be added to $\mathscr{C}_1$. Following Lemma 1, π_1, a sequence consisting of a_j^1 copies of c_j^1 $(1 \leqslant j \leqslant i_1)$, can be constructed. Now suppose $\mu_1 = \mu_0 + \Delta(\pi_1)$, and $\sigma_1 = \sigma \dot{-} \pi_1$. Guaranteed by the claim stated in the beginning of the proof, there exists a rearrangement σ_1' of σ_1 such that $\mu_1 \xmapsto{\sigma_1'}$. If σ_1' is circuit-free, we are done; otherwise, let σ_1'' be the *shortest* prefix of σ_1' such that the remaining sequence $\sigma_1' \dot{-} \sigma_1''$ covers a circuit, say $\bar{c}$, which is not token-free with respect to μ_2, where $\mu_1 \xmapsto{\sigma_1''} \mu_2$. (Notice that it is possible for a $\bar{c}$ to be marked in μ_1; in this case, σ_1'' is empty.) Since σ_1'' is circuit-free, and each transition in σ_1'' removes at most one token from a place, σ_1'' can be rearranged into $\alpha_1\alpha_1'$ in such a way that ($\forall t \in \mathrm{T}$, $\#_{\alpha_1}(t) \leqslant 1$), $(\Delta(\alpha_1)(p) \leqslant 1, \forall p \in P)$, and $(\mu_2'(\bar{c}) > 0)$, where $\mu_1 \xmapsto{\alpha_1} \mu_2'$. (Intuitively, α_1 is a simple path (in the graph-theoretic sense) leading to some place in $\bar{c}$.) The remaining π_i and α_i can be constructed similarly. $\square$

Using Lemma 2, we are able to set up a system of linear inequalities to capture reachability for BPP-nets, giving rise to an NP upper bound for the reachability problem since *integer linear programming* is in NP. Before doing so, we require the following known result from [16].

Lemma 3 (Yamasaki [16, Theorem 3.3]). *If a PN $\mathscr{P} = ((P, T, \varphi), \mu_0)$ has no token-free circuits in every reachable marking, then $R(\mathscr{P}) = \{\mu | \mu = \mu_0 + A \cdot x \geqslant 0$, for some $x \in N^m\}$, where m is the number of transitions in T, and A is the addition matrix. In words, $\mu = \mu_0 + A \cdot x \geqslant 0$ is a sufficient and necessary condition for reachability provided that no token-free circuit is reachable in the PN.*

As a direct consequence, $\mu = \mu_0 + A \cdot x \geqslant 0$ is also a sufficient and necessary condition for reachability for circuit-free PNs.

Theorem 4. *The reachability problem for BPP-nets can be solved in* NP.

Proof. As shown in Lemma 2, μ is reachable from the initial marking μ_0 iff there exists a sequence $\sigma=\pi_1\alpha_1\pi_2\alpha_2\cdots\pi_h\alpha_h$ ($\mu_0 \xmapsto{\sigma} \mu$) meeting the three conditions stated in Lemma 2.

The desired system of linear inequalities can be set up as follows:

1. For $1\leqslant i\leqslant h$, guess $\mathscr{C}_i$ $(=\{c_1^i,\ldots,c_{r_i}^i\})$ and verify the connectivity condition; for $1\leqslant i\leqslant h-1$, guess the sequence α_i and check Conditions 2(a)–(c) of Lemma 2; guess the set $\{t_{h_1},\ldots,t_{h_l}\}$ of transitions used in α_h and verify the circuit-freedom condition. It is not hard to see that checking each of the above can be done in polynomial time.

2. Let μ_i and μ_i' $(\geqslant \mathbf{0})$ be marking variables, and a_j^i and b_j be scalar variables carrying positive integer values. Set up the following linear inequalities to capture PN computation $\mu_0=\mu_1 \xmapsto{\pi_1} \mu_1' \xmapsto{\alpha_1} \mu_2 \xmapsto{\pi_2} \mu_2' \xmapsto{\alpha_2} \cdots \mu_h \xmapsto{\pi_h} \mu_h' \xmapsto{\alpha_h} \mu$:

$$\mu_1=\mu_0 \tag{1}$$

$$\mu_i'=\mu_i+\sum_{j=1}^{r_i} a_j^i * \Delta(c_j^i), \quad \forall 1\leqslant i\leqslant h, \tag{2}$$

$$\mu_{i+1}=\mu_i'+\Delta(\alpha_i), \quad \forall 1\leqslant i\leqslant h-1, \tag{3}$$

$$\mu=\mu_h'+\sum_{j=1}^{l} b_j * \Delta(t_{h_j}), \tag{4}$$

(1) is trivial. For (2), the validity of μ_i' being reachable from μ_i is guaranteed by Lemma 1. Since α_i $(1\leqslant i<h)$ is circuit-free, (3) is sufficient to ensure the reachability of μ_{i+1} from μ_i' through the firing of α_i, as Lemma 3 suggests. Likewise, circuit-freedom of α_h justifies (4). (Notice that the need to consider α_h separately comes from the fact that α_h may contain multiple copies of a transition. See Lemma 2.)

In view of the above, μ is reachable from μ_0 iff the above system of linear inequalities has integer solutions with respect to variables μ_i, μ_i', a_j^i and b_j. This completes the proof of the theorem. □

The NP upper bound of testing reachability for BPP-nets was first shown in [4], providing a simpler proof for the NP upper bound of the *uniform word problem* for *context-free commutative grammars*, which was originally shown in [10]. (The concept of the so-called *siphon* plays a crucial role in the proof of [4].) By providing a new characterization for paths in BPP-nets, we offer yet another approach to solving the reachability problem for BPP-nets.

4. The equivalence problem for BPP-nets

In this section, we investigate the complexity of the equivalence problem for BPP-nets. Our upper bound is obtained by demonstrating 'small' semilinear set representations for the reachability sets of BPP-nets. More precisely, we have the following theorem.

Theorem 5. *Let $\mathscr{P} = ((P, T, \varphi), \mu_0)$ be a BPP-net of size s. For some fixed constants c_1, c_2, d_1, d_2, d_3 independent of s, we can construct in DTIME$(2^{c_2 s^3})$ a semilinear reachability set $R(\mathscr{P}) = \bigcup_{v \in B} \mathscr{L}(v, \rho_v)$ (whose size is bounded by $O(2^{c_1 s^3})$), where*

1. *B is the set of all reachable markings with no component larger than $2^{d_1 s^2}$, and*
2. *ρ_v is the set of all $\vartheta \in N^k$ such that*
   (a) *ϑ has no component larger than $2^{d_2 s^2}$, and*
   (b) *$\exists\ \sigma, \sigma_1, \sigma_2 \in T^*$, $\exists$ marking μ_1,*
   (i) $\mu_0 \overset{\sigma_1}{\longmapsto} \mu_1 \overset{\sigma_2}{\longmapsto} v$,
   (ii) $\mu_1 \overset{\sigma}{\longmapsto} \mu_1 + \vartheta$,
   (iii) $|\sigma|, |\sigma_1 \sigma_2| \leqslant 2^{d_3 s^2}$.

Proof. Let $\mathscr{P}$ be of k places and m transitions, and n be a number such that $\mu_0(p) \leqslant n$ and $|\varphi(t, p)| \leqslant n$, $\forall t \in T, p \in P$, i.e., no integer mentioned in $\mathscr{P}$ is larger than n. (Recall that for BPP-nets, it is always the case that $|\varphi(p, t)| \leqslant 1$, $\forall t \in T, p \in P$.) Clearly, $m, k \leqslant s$ and $n \leqslant 2^s$.

$\bigcup_{v \in B} \mathscr{L}(v, \rho_v) \subseteq R(\mathscr{P})$ is obvious, since, according to Condition 2(b)ii, σ can be pumped in marking μ_1 for an arbitrary number of times. Therefore, it is sufficient to show $R(\mathscr{P}) \subseteq \bigcup_{v \in B} \mathscr{L}(v, \rho_v)$. The proof is somewhat involved. To better explain the details, Fig, 5 illustrates the key steps of the proof. The reader is encouraged to consult Fig. 5 as our discussion progresses.

Let $\mu \in R(\mathscr{P})$ be a reachable marking. According to Lemma 2, there exists a sequence $\pi_1 \alpha_1 \pi_2 \alpha_2 \cdots \pi_h \alpha_h$ which witnesses $\mu_0 \overset{*}{\longmapsto} \mu$, and satisfies Conditions (1)–(3) stated in the description of Lemma 2. See Fig. 5(a). For ease of explanation, let $\delta' = \pi_1 \alpha_1 \pi_2 \alpha_2 \cdots \pi_h$, $\delta = \alpha_h$, and $\mu_0 \overset{\delta'}{\longmapsto} \mu' \overset{\delta}{\longmapsto} \mu$, for some μ'. Recall that $\Delta(\pi_i) = \sum_{j=1}^{r_i} a_j^i \Delta(c_j^i)$ for some integers $a_1^i, a_2^i, \ldots, a_{r_i}^i > 0$ (see Lemma 2). (In Fig. 5, for example, π_1 consists of three circuits c_1, c_2, and c_3 of multiplicities 60, 50 and 10, respectively.) Consider circuits in π_i, $1 \leqslant i \leqslant h$. It is clear from Condition 2(b) of Lemma 2 that for any place p, each of $\alpha_i, \ldots, \alpha_{h-1}$ consumes at most one token from p; hence, the entire sequence $\alpha_i, \ldots, \alpha_{h-1}$ consumes at most $h - i$ tokens from p. Now if $a_j^i > m (\geqslant h - i)$, for some j, $1 \leqslant j \leqslant r_i$, then $\mu_0 \overset{\delta' \dot{-} c_j^i}{\longmapsto} (\mu' - \Delta(c_j^i))$ remains a valid path. In words, a copy of circuit c_j^i can be cut without rendering the path invalid. (This is mainly because if the firing of $\alpha_i, \ldots, \alpha_{h-1}$ hinders on circuit c_j^i, m copies of c_j^is suffice.) By trimming excess copies of circuits in δ' repeatedly (called the resulting sequence δ''), we have $\mu_0 \overset{\delta''}{\longmapsto} \mu''$, for some μ'', such that no circuit in δ'' appears more than m times, and $\mu'' + \sum_{\pi \in Q} \Delta(\pi) = \mu'$, where Q is a *multiset* containing those circuits cut in the above trimming process. In Fig. 5(b), m is assumed to be 10; hence, 50 copies of c_1 and 40 copies of c_2 are thrown into Q. It is important to note that for every π in Q, π is enabled in some $\bar{\mu}$ in $\overbrace{\mu_0 \overset{*}{\longmapsto} \bar{\mu} \overset{*}{\longmapsto} \mu''}^{\delta''}$. Now consider the

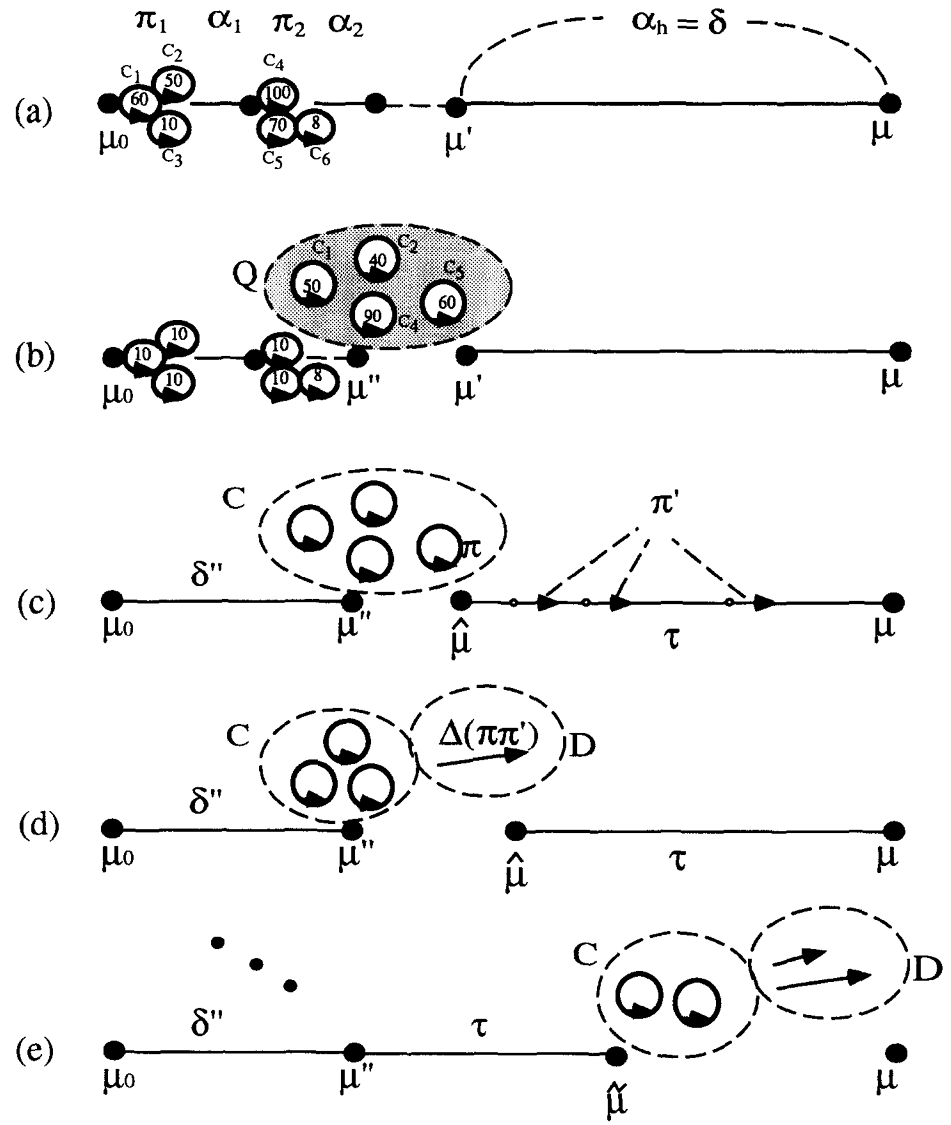

Fig. 5. The construction of a semilinear set representation of a BPP-net.

length of δ''. According to Condition 1(c) of Lemma 2, the number of distinct circuits 'collected' along δ'' is bounded by m. Furthermore, the above discussion suggests that there are at most m copies in existence for each such circuit. Hence, the sum of the lengths of all such circuits is bounded by m^3 (there are at most m^2 circuits, each of which is of length $\leqslant m$). In addition, the length of $\alpha_1 \cdots \alpha_{h-1}$ is bounded by $(h-1)*m$ ($\leqslant m^2$). As a consequence, $|\delta''| \leqslant m^3 + m^2 (\leqslant 2m^3)$; hence, no component in μ'' exceeds $n + 2m^3 n$ (i.e., an upper bound on the size of the initial marking + an upper bound on the number of tokens that can be deposited into a place resulting from firing δ'') $= O(m^3 n)$.

Now consider $\mu' \xmapsto{\delta} \mu$, which covers no circuits. Unlike $\alpha_1, \ldots, \alpha_{h-1}$ (all of which are short), in δ ($=\alpha_h$) the number of times a transition is used may not be bounded by a polynomial. In our subsequent discussion, we show how pieces of the suffix δ (if it is too long) can be paired with circuits in Q, resulting in a sufficiently short suffix. Upon the completion of this 'pairing' process, the base and periods of the semilinear set follow immediately. Such a construction is done in an iterative fashion. Initially, let variables C (a multiset of circuits), D (a multiset of vectors), $\hat{\mu}$ (a marking, which can also be viewed as a vector), and τ (a transition sequence) be Q, $\emptyset$, μ', δ, respectively. The key of our construction relies on proving that the following remains an invariant as the iteration progresses:

$$\bullet(\mu_0 \xmapsto{\delta''} \mu'') \wedge \left(\mu'' + \left(\sum_{\pi \in C} \Delta(\pi)\right) + \left(\sum_{\theta \in D} \theta\right) = \hat{\mu}\right) \wedge (\hat{\mu} \xmapsto{\tau} \mu).$$

To illustrate the intuition, consider Fig. 5(c). We show that if τ (which equals δ initially) is 'too long', then it must contain a short segment π' (see Fig. 5(c)) that can be paired with some circuit π in C (see Fig. 5(d)) in such a way that $\Delta(\pi\pi')$ remains nonnegative. (Hence, $\Delta(\pi\pi')$ can then be placed into the final set of periods.) By repeatedly doing so, τ can be shortened.

Clearly, the invariant holds initially. In what follows, we explain in detail how the above intuition of constructing the semilinear set (in an iterative fashion) is implemented. Recall that an ordering is assumed on the elements of P and T, (i.e., $P = \{p_1, \ldots, p_k\}$ and $T = \{t_1, \ldots, t_m\}$). Let A be a $k \times m$ *addition matrix* of (P, T, φ) so that $a_{i,j} = \varphi(t_j, p_i) - \varphi(p_i, t_j)$. Now let π be a circuit in C. Consider the following optimization formula (in which $x_1, \ldots, x_m$ are nonnegative variables):

$$\text{maximize } \sum_{i=1}^{m} x_i$$

$$\text{subject to} \begin{cases} \Delta(\pi) + A \cdot (x_1, \ldots, x_m)^{\mathrm{T}} \geqslant \mathbf{0}, \\ (x_1, \ldots, x_m) \leqslant \#_\tau. \end{cases}$$

In words, solution $X = (x_1, \ldots, x_m)$ represents the transition count vector of the maximum firable sequence contained in τ using only tokens accumulated as a result of firing circuit π. Consider two cases:

Case 1: $X \neq \mathbf{0}$, for some π in C. Since τ is circuit-free, guaranteed by Lemma 3 there exists a π' such that $\#_{\pi'} = (x_1, \ldots, x_m)$ and $\pi\pi'$ is enabled in some $\bar{\mu}$ in $\overbrace{\mu_0 \xmapsto{*} \bar{\mu} \xmapsto{*} \mu''}^{\delta''}$. Let $\theta = \Delta(\pi\pi') = \Delta(\pi) + A \cdot X^T \geqslant \mathbf{0}$. Since $(\Delta(\pi)(p_i) \leqslant mn, \forall p_i \in$ P), the maximum number of tokens that can pile up in a place using a circuit-free transition sequence is bounded by $(mn)n^{l-1}$, where $l = min\{m, k-1\}$. (The worst-case scenario arises when π' is a path $p_i t_{g_1} p_{g_1} \cdots p_{g_{j-1}} t_{g_j} p_{g_j} \cdots t_{g_l} p_{g_l}$ along which each transition t_{g_j} 'amplifies' the token count of its input place by a factor of n.) Hence, $(\theta(i) = \Delta(\pi\pi')(p_i) \leqslant mn + (mn)n^{l-1} \leqslant 2mn^l$ $(\leqslant 2^{d_2 s^2}$, for some constant d_2), $\forall p_i \in P$), and $|\pi\pi'| \leqslant \overbrace{m}^{\geqslant|\pi|} + \overbrace{mn + mn^2 + mn^3 + \cdots + mn^l}^{\geqslant|\pi'|} \leqslant (l+1)mn^l$. We let $D := D \cup \{\theta\}$, $C := C - \{\pi\}$, $\hat{\mu} := \hat{\mu} + \Delta(\pi')$, and $\tau :=$ a firable sequence (in $\hat{\mu}$) using the transitions from $\tau \dot{-} \pi'$. See Figure 5(d). (Notice that $\hat{\mu} + \Delta(\tau) = \mu$ and τ being circuit-free imply the existence of such a firable sequence (Lemma 3)). Clearly, the invariant remains true.

Case 2: The above iteration ends when $X = \mathbf{0}$, for *every* π in C, or $C = \emptyset$. We claim that $\mu'' \xmapsto{\tau} \tilde{\mu}$, for some $\tilde{\mu}$. If this is the case, by letting the base (v) and the set (ρ_v) of the periods of the semilinear set be $\tilde{\mu}$ and $\{\Delta(\pi) | \pi \in C\} \cup D$, respectively, we have $\mu_0 \xmapsto{\delta''\tau} \tilde{\mu}$, and $\tilde{\mu} + (\sum_{\pi\in C} \Delta(\pi)) + (\sum_{\theta\in D} \theta) = \mu$. Furthermore, in our earlier discussion we know that $\forall p_i \in P$, $\mu''(p_i) \leqslant n + 2m^3n \leqslant 3m^3n$. Hence, $|\tau| \leqslant 3m^3n + 3m^3n^2 + \cdots + 3m^3n^l \leqslant 3lm^3n^l$, and $\tilde{\mu}(p_i) \leqslant (3m^3n)n^{l-1}$ $(\leqslant 2^{d_1 s^2}$, for some constant d_1), since τ is circuit-free. See Fig. 5(e). Now we prove the claim (i.e., $\mu'' \xmapsto{\tau} \tilde{\mu}$). If the firing of τ in μ'' fails for some transition t in τ, then it must be the case that some circuit in C provides a token for t in the original path – contradicting the fact that in (Case 2), either the solution to the above optimization problem is $\mathbf{0}$ or C is empty. (Notice that the token needed by t cannot come from any $\pi\pi'$ that has already been added to D; otherwise, π' violates the requirement of being the maximum solution.) By picking a constant d_3 so that $|\pi\pi'| \leqslant (l+1)mn^l \leqslant 2^{d_3 s^2}$ and $\delta''\tau \leqslant (n + 2m^3n) + 3m^3n^l \leqslant 2^{d_3 s^2}$, Condition 2(b)iii holds.

Now we show the size of the semilinear set representation as well as the time required for generating such a representation. From Condition 1, each component of v is bounded by $2^{d_1 s^2}$; hence $|B|$ (i.e., the number of distinct bases of the semilinear set) is bounded by $(2^{d_1 s^2})^k \leqslant 2^{d_1 s^3}$ ($k (\leqslant s)$ is the dimension of v). As a result, the size of B is bounded by $k * (\log_2(2^{d_1 s^2})) * (2^{d_1 s^3})$. Likewise, from Condition 2 the size of each period ρ_v is bounded by $k * (\log_2(2^{d_2 s^2})) * (2^{d_2 s^3})$. In summary, the size of $\bigcup_{v\in B} \mathscr{L}(v, \rho_v)$ is (the size of B) + $\sum_{v\in B}$(the size of ρ_v), which is bounded by $O(2^{c_1 s^3})$, for some constant c_1. As for the amount of time needed to generate the semilinear set, first recall from Theorem 4 that the reachability problem for BPP-nets in NP. From our earlier discussion, each base vector v is of size $k * (\log_2(2^{d_2 s^2}))$, which is polynomial in $O(s^3)$. Hence, the reachability of v from μ_0 can be checked in $DTIME(2^{O(s^3)})$. Similarly, checking the existence of a positive loop satisfying Conditions 2(a) and (b) can also be done in $DTIME(2^{O(s^3)})$. By exhaustive search, the desired semilinear set can be constructed in $DTIME(2^{c_2 s^3})$, for some constant c_2. $\square$

To show our main result, we also require the following known result concerning the complexity of the equivalence problem for semilinear sets (see [9, 12]):

Lemma 6 (Huynh [9, Corollary 5.2]). *The equivalence problem for semilinear sets is in Π_2^P.*

From Theorem 5 and Lemma 6, we immediately have:

Corollary 7. *The equivalence problem for BPP-nets is solvable in* $DTIME(2^{2^{d*s^3}})$, *where s is the size of the PN, and d is some fixed constant.*

As for the lower bound, it is known that the equivalence problem for commutative context-free grammars is Π_2^P-hard (see [11]). Since commutative context-free grammars are a special case of BPP-nets, the following lower bound for BPP-nets follows immediately.

Theorem 8. *The equivalence problem for BPP-nets is* Π_2^P*-hard.*

Acknowledgements

The author thanks the anonymous referees for their comments and suggestions which improved the presentation as well as the correctness of this paper.

References

[1] S. Christensen, Decidability and decomposition in process algebras, Ph.D. Thesis, Univ. of Edinburgh, UK, CST-105-93, 1993.

[2] S. Christensen, Y. Hirshfeld and F. Moller, Bisimulation equivalence is decidable for basic parallel processes, in: *Proc. CONCUR'93*, Lecture Notes in Computer Science, Vol. 715 (Springer, Berlin, 1993) 143–157.

[3] S. Christensen and H. Hüttel, Decidability issues for infinite-state processes – a survey, *Bull. EATCS* **51** (1993) 156–166.

[4] J. Esparza, Petri nets, commutative context-free grammars and basic parallel processes, in: *Proc. FCT'95*, Lecture Notes in Computer Science, Vol. 965 (Springer, Berlin, 1995) 221–232.

[5] J. Esparza and M. Nielsen, Decidability issues for Petri nets – a survey, *Bull. EATCS* **52** (1993) 245–262.

[6] Y. Hirshfeld, Petri nets and the equivalence problem, in: *Proc. CSL'93, 1993 Conf. of European Association for Computer Science Logic*, Lecture Notes in Computer Science, Vol. 832 (Springer, Berlin, 1994) 165–174.

[7] Y. Hirshfeld, M. Jerrum and F. Moller, A polynomial time algorithm for deciding bisimulation equivalence of normed basic parallel processes, Tech. Report, Dept. of Computer Science, Univ. of Edinburgh, UK, 1994.

[8] H. Hüttel, Undecidable equivalences for basic parallel processes, in: *Proc. Int'l Symp. on Theoretical Aspects of Computer Software*, Lecture Notes in Computer Science, Vol. 789 (Springer, Berlin, 1994) 454–464.

[9] D. Huynh, The complexity of semilinear sets, *Inform. Process. Cybernet. (EIK)* **18** (1982) 291–338.

[10] D. Huynh, Commutative grammars: the complexity of uniform word problems, *Inform. Comput.* **57** (1983) 21–39.

[11] D. Huynh, The complexity of equivalence problems for commutative grammars, *Inform. Comput.* **66** (1985) 103–121.

[12] D. Huynh, A simple proof for the Σ_2^P upper bound of the inequivalence problem for semilinear sets, *Inform. Process. Cybernet.* (*EIK*) **22** (1986) 147–156.

[13] J. Peterson, *Petri Net Theory and the Modeling of Systems* (Prentice-Hall, Englewood Cliffs, NJ, 1981).

[14] W. Reisig, *Petri Nets: An Introduction*, EATCS Monographs in Computer Science, Vol. 4 (Springer, Berlin, 1985).

[15] L. Stockmeyer, The polynomial-time hierarchy, *Theoret. Comput. Sci.* **3** (1977) 1–22.

[16] H. Yamasaki, Normal Petri nets, *Theoret. Comput. Sci.* **31** (1984) 307–315.

[17] R. van Glabbeek, The linear time – branching time spectrum, in: *Proc. CONCUR'90*, Lecture Notes in Computer Science, Vol. 458 (Springer, Berlin, 1990) 278–297.